**Computer Vision for Text Detection** **From Pixels to Text: Comparing OCR Engines on Complex Image Data**

|  |  |
| --- | --- |
| **Jelvinlon Brilliando\***  Computer Science Department,  School of Computer Science,  Bina Nusantara University  Bandung Campus, Jakarta,  Indonesia 11480  [Jelvinlon.brilliando@binus.ac.id](mailto:Jelvinlon.brilliando@binus.ac.id)  **Muhammad Robby Kusumah**  Computer Science Department,  School of Computer Science,  Bina Nusantara University  Bandung Campus, Jakarta,  Indonesia 11480  [muhammad.kusumah001@binus.ac.id](mailto:muhammad.kusumah001@binus.ac.id) | **Riccosan**  Computer Science Department,  School of Computer Science,  Bina Nusantara University  Bandung Campus, Jakarta,  Indonesia 11480  [riccosan@binus.ac.id](mailto:riccosan@binus.ac.id) |

***Abstract*** *–Optical Character Recognition (OCR) is a critical technology for converting image-based text into machine-readable data. While a variety of OCR tools are available, identifying the most effective engine for challenging conditions—such as real-world scene text—remains a significant challenge due to factors like cluttered backgrounds, variable lighting, and diverse fonts.This study provides a direct, quantitative comparison of three popular open-source OCR libraries: EasyOCR, Keras-OCR, and Pytesseract (Tesseract). The evaluation was performed using a random sample of 100 images drawn from the diverse TextOCR dataset, which contains incidental text embedded in real-world scenes. Accuracy was measured by comparing each model’s output against verified ground truth using the Levenshtein Ratio, a metric that quantifies the similarity between predicted and actual text.The results revealed a notable performance gap between traditional and deep learning-based models. Keras-OCR emerged as the top performer, achieving an average accuracy of 47.26%, followed closely by EasyOCR at 46.52%. In stark contrast, Pytesseract, which leverages the classical Tesseract engine, achieved only 8.94% accuracy, frequently failing to detect text in complex, natural scenes.These findings strongly support the use of modern deep learning–based OCR frameworks like Keras-OCR and EasyOCR for scene text recognition tasks. Their superior performance underlines their suitability for applications that involve extracting text from uncontrolled, real-world image environments—providing valuable guidance for developers and researchers working in computer vision and document analysis.*

***Keywords*** *–* Computer Vision, Text Detection, Optical Character Recognition (OCR), EasyOCR, Tesseract OCR, Keras-OCR, Deep Learning, Scene Text Recognition, TextOCR Dataset, Benchmarking OCR Models

1. **INTRODUCTION**

In the digital age, the capability to effectively extract and interpret text from images and scanned documents has become ever more vital [1]. The increase of handwritten and printed text in different formats, such as invoices, legal documents, signboards, and manuscripts, requires strong methods for automated text detection [2]. Traditional manual text extraction methods are slow, prone to errors, and labor-heavy, especially when handling large quantities of documents [3]. As a result, there is an escalating need for intelligent systems capable of conducting text detection with high precision and efficiency.

Text detection is an essential element of Optical Character Recognition (OCR) and is crucial for document digitization, content indexing, and improving accessibility [4]. However, obstacles like variations in font styles, text orientations, low contrast, and background noise make the accuracy of standard OCR systems more difficult [4]. Recent developments in deep learning, particularly within computer vision, have transformed text detection by utilizing convolutional neural networks (CNNs) and transformer-based frameworks [5]. These models have showcased superior capabilities in identifying text across various settings, including images from natural scenes and historical manuscripts [6].

Benchmarking OCR Models,This research benchmarks EasyOCR against other commonly utilized OCR models, such as Tesseract and Keras-OCR, utilizing the TextOCR dataset [7]. The benchmarking procedure includes Evaluation Metrics: Assessing accuracy, processing speed, memory usage, and reliability across varying image qualities.Dataset Selection: Utilizing the TextOCR dataset, which features over one million word annotations from natural scene images, ensuring a range of real-world testing circumstances. Experiments: Each OCR model undergoes testing on the identical dataset under the same conditions to evaluate comparative performance.

Results Analysis: The results will showcase the advantages and disadvantages of each OCR model, offering insights into the most suitable applications for different situations. Through the comparison of these models, this study seeks to identify the most efficient and precise OCR system for diverse document processing requirements, providing valuable suggestions for industry [8]. Dataset Selection: Utilizing the TextOCR dataset, which features over one million word annotations from natural scene images, ensuring a range of real-world testing circumstances.Experiments: Each OCR model undergoes testing on the identical dataset under the same conditions to evaluate comparative performance.Results Analysis: The results will showcase the advantages and disadvantages of each OCR model, offering insights into the most suitable applications for different situations.Through the comparison of these models, this study seeks to identify the most efficient and precise OCR system for diverse document processing requirements, providing valuable suggestions for industry.

1. **LITERATURE REVIEW**

This literature review aims to map out existing approaches and techniques used in previous studies related to computer vision for text detection, including the algorithms, datasets, and performance metrics commonly employed. Furthermore, this review identifies the limitations and gaps in current research, providing a foundation for the contribution of the present study toward developing a more robust and efficient document automation system.

Shaw and Mishra in 2025 introduce a machine learning-based OCR model specifically designed to enhance character detection and recognition of English text across diverse sources, including real-time videos, objects, and static images. Their system incorporates a self-optimizing mechanism aimed at improving accuracy and efficiency in converting printed or displayed English text into editable digital formats. Targeted applications include digital libraries, text analysis, and information retrieval systems, with the model achieving a notable 96% recognition rate on English alphabet and numeric characters. The study emphasizes how improved OCR clarity can support users who face challenges in interpreting or interacting with text-based content [9].

While the model demonstrates promising accuracy in English character recognition, the research does not explore comparisons with other mainstream OCR systems such as Tesseract, EasyOCR, or Keras-OCR. Moreover, its applicability is confined to the English language, with limited insights into multilingual or layout-diverse document environments. As such, while the study contributes to improving OCR precision for a specific linguistic and use-case context, it does not address broader benchmarking challenges or evaluate model performance across varying document complexities and real-world datasets [9].

The next study reviewed is by Eken, Menhour, and Köksal in 2019, who introduced a content-based automatic classification framework known as DoCA (Document Classification and Analysis) [10]. This system was designed to simplify and automate the classification of digital files across various formats, including office documents, scanned images, and multimedia files. The framework applies different preprocessing and analysis strategies tailored to each file type, offering a flexible and scalable approach to document automation. Its implementation reflects the increasing need for efficient handling of heterogeneous digital content in organizational and institutional settings [10].

Although DoCA demonstrates promising performance, as evaluated on the HAVELSAN dataset, the paper does not provide detailed quantitative results in the abstract. Furthermore, while the system effectively addresses structured and semi-structured documents, it does not focus on scene text detection, which limits its applicability in environments where unstructured or natural image-based text is prevalent. Nevertheless, DoCA serves as a relevant foundation for understanding multi-format document analysis and highlights the importance of adaptive classification mechanisms in automated document processing systems.

Study by Khallouli et al. In 2024 focus on the unique challenges of digitizing legacy engineering documents, which often exist in non-digital formats like scanned PDFs or printed 2D drawings. Recognizing the limitations of conventional OCR tools in handling such documents, the authors propose a transformer-based OCR system enhanced by generative data augmentation and transfer learning techniques. Their model is tailored to recognize text within complex engineering drawings, a domain that poses difficulties due to its diverse layouts and specialized terminology. Evaluated on a dataset comprising ship engineering documents, the proposed system significantly outperformed standard pretrained OCR models, showcasing the strength of transformer architectures in specialized document contexts. This study highlights the potential of modern deep learning techniques in improving OCR performance for technical and domain-specific documents, although it does not provide a comparative analysis across multiple mainstream OCR systems or generalize its findings to more diverse real-world datasets [4].

Beshirov et al. In 2025 address the challenges of digitizing historical Bulgarian documents by focusing on post-OCR text correction [11]. While OCR is a pivotal step in preserving cultural heritage through digitization, standard OCR tools often struggle with the non-standardized orthography and complex layouts found in historical texts. To tackle this issue, the authors introduce the first benchmark dataset for evaluating OCR text correction in Bulgarian historical documents written in the Drinov orthography of the 19th century. They also propose a method for generating synthetic data in both Drinov and Ivanchev orthographies by transforming modern Bulgarian texts. This allows for robust training data even in the absence of extensive historical corpora [11].

Their correction method utilizes a combination of encoder-decoder frameworks with enhancements such as diagonal attention loss, copy mechanisms, and coverage mechanisms. These augmentations significantly improve post-OCR correction performance, yielding a 25% improvement in document quality, outperforming the state-of-the-art by 16% on the ICDAR 2019 Bulgarian dataset. However, this study is heavily focused on language-specific historical text correction and does not benchmark or compare OCR engine performance directly. While it demonstrates strong results in error correction, it does not offer insights into OCR model selection, speed, or resource efficiency across modern OCR frameworks—a gap that benchmarking studies, such as the one using EasyOCR, Tesseract, and Keras-OCR, aim to address [12].

Soni et al. investigate the challenges of scene text detection and recognition in natural images by leveraging a fusion of the EAST (Efficient and Accurate Scene Text Detector) algorithm with multiple OCR engines, namely TesseractOCR, PaddleOCR, and EasyOCR [12]. The study emphasizes the complexities involved in detecting and recognizing texts in real-world conditions, where variations in font, orientation, color, and background interference pose significant hurdles. The authors focus on both Scene Text Detection (STD) and Scene Text Recognition (STR), implementing the EAST model for detecting text regions and then evaluating the OCR performance based on efficiency and recognition accuracy. The integration of detection and recognition steps provides a practical framework for evaluating OCR models in a structured and measurable way [11].

This research provides empirical insights by analyzing inference times and detection outcomes for each OCR engine. The average inference times demonstrate EAST’s ability to operate efficiently in real-time applications [13]. While this study aligns closely with the goals of OCR benchmarking, it focuses more on detection-then-recognition pipelines in natural scenes rather than full benchmarking of OCR models on large-scale annotated datasets. In contrast, my study benchmarks OCR models using the comprehensive TextOCR dataset to assess accuracy, speed, and memory usage across a broader set of real-world image conditions, without the dependency on a specific detector. This highlights a gap in detection-agnostic benchmarking approaches, where OCR models are tested independently for their full-text recognition performance, offering complementary insights to detection-fusion methods [11].

Based on the reviewed literature, it is evident that significant strides have been made in the development of OCR technologies and document automation systems [14]. Prior studies have explored diverse techniques ranging from traditional machine learning and rule-based approaches to advanced deep learning architectures, including transformer-based models and end-to-end scene text detectors [15]. These approaches have been tailored to specific contexts such as real-time video analysis, legacy engineering documents, historical texts, and heterogeneous digital formats. Notable contributions include improvements in recognition accuracy, layout adaptability, and domain-specific customization [16]. However, most of these studies either focus on narrow use-case scenarios, specific languages, or fail to provide comparative evaluations of multiple OCR engines under standardized benchmarking conditions.

This review highlights a recurring gap in the field: the lack of comprehensive benchmarking studies that evaluate OCR models holistically, across multilingual datasets, diverse text layouts, and varying levels of image complexity. While some research emphasizes post-OCR correction or document classification, others prioritize detection-recognition pipelines without isolating OCR model capabilities. Thus, there remains a need for detection-agnostic benchmarking that rigorously tests OCR engines based on accuracy, inference time, and memory efficiency across large-scale, real-world datasets. The present study aims to address this gap by systematically evaluating the performance of multiple OCR frameworks using the TextOCR dataset, contributing to the development of more robust and efficient document automation systems applicable across a wide range of practical settings.

1. **METHODOLOGY**

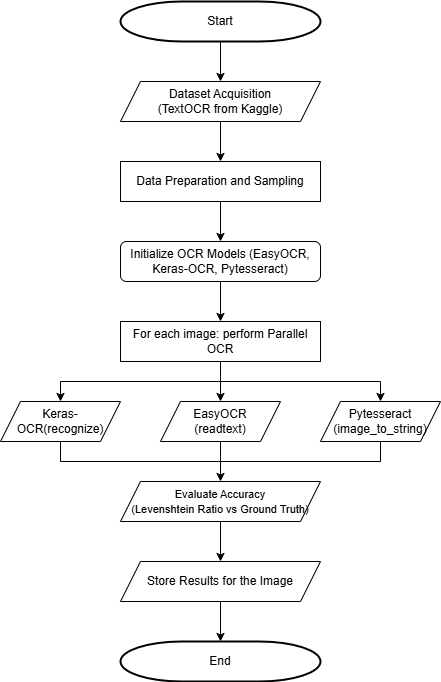


Figure 1.Workflow

The methodology for this comparative study was designed to ensure a reproducible and fair evaluation of three prominent Optical Character Recognition (OCR) models. The entire process, from data acquisition to final analysis, is visually summarized in Figure 1.

**3.1. Dataset**

This study utilized the TextOCR dataset, a large-scale collection of real-world images with incidental text, sourced from the Kaggle repository(Singh et al., 2021). The dataset is notable for its diversity of text appearances, including various fonts, scales, and lighting conditions, making it a challenging and realistic benchmark. From the full training set, a random sample of 300 images was selected for the analysis to ensure a manageable computational workload. The ground truth for each image, consisting of the correct text transcription, was extracted from the provided TextOCR\_0.1\_train.json annotation file.

**3.2*.*****Experimental Setup**

The experiment was conducted within the Google Colaboratory cloud environment, which provided access to a GPU for accelerating model inference. The analysis was scripted in Python 3. The primary libraries used for the OCR comparison were EasyOCR, Keras-OCR, and Pytesseract. Data manipulation and storage were handled using the pandas library, and the python-levenshtein library was used for accuracy calculations.

**3.3. OCR Models and Technologies**

Three distinct OCR models were selected for evaluation based on their popularity and different underlying architectures such as EasyOCR: A popular open-source library designed for ease of use and support for multiple languages. It employs a deep learning-based approach for both text detection and recognition [17]. Keras-OCR: A library providing a pre-trained OCR pipeline. For its text detection stage, this library implements the highly effective Character Region Awareness for Text Detection (CRAFT) model [18]. Text recognition is subsequently handled by a Convolutional Recurrent Neural Network (CRNN). Pytesseract is a Python wrapper for Google's Tesseract OCR engine, one of the most widely known open-source OCR engines. Tesseract has undergone significant evolution, with recent versions incorporating Long Short-Term Memory (LSTM) network models for improved accuracy [19].

**3.4. Evaluation Metric**

To quantitatively measure the performance of each OCR model, the Levenshtein Ratio was employed as the primary evaluation metric. The Levenshtein Ratio calculates the similarity between two strings—the ground truth text and the model's predicted text—as a percentage score from 0% to 100% [20]. It is derived from the Levenshtein distance, which counts the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one string into the other. A higher ratio signifies a more accurate transcription. The final reported accuracy for each model is the arithmetic mean of the Levenshtein Ratios across all 300 processed images [21].

1. **RESULTS**

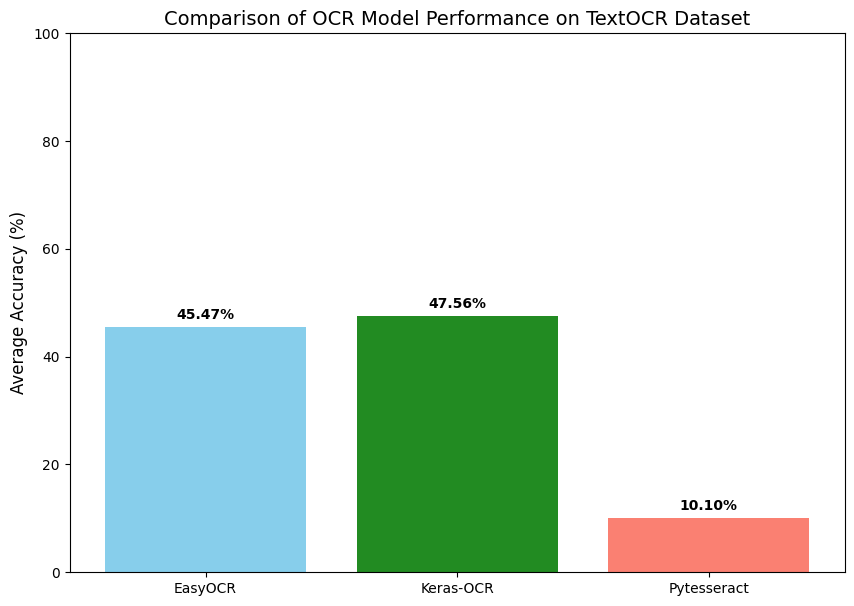
This section presents the objective findings from the experiment, detailing both the quantitative performance and qualitative examples.

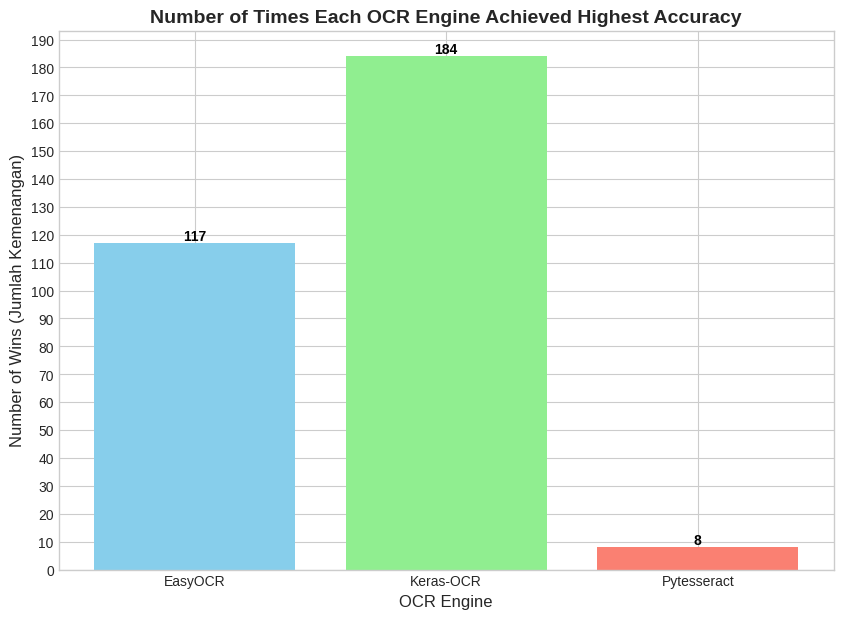
**4.1. Quantitative Analysis**

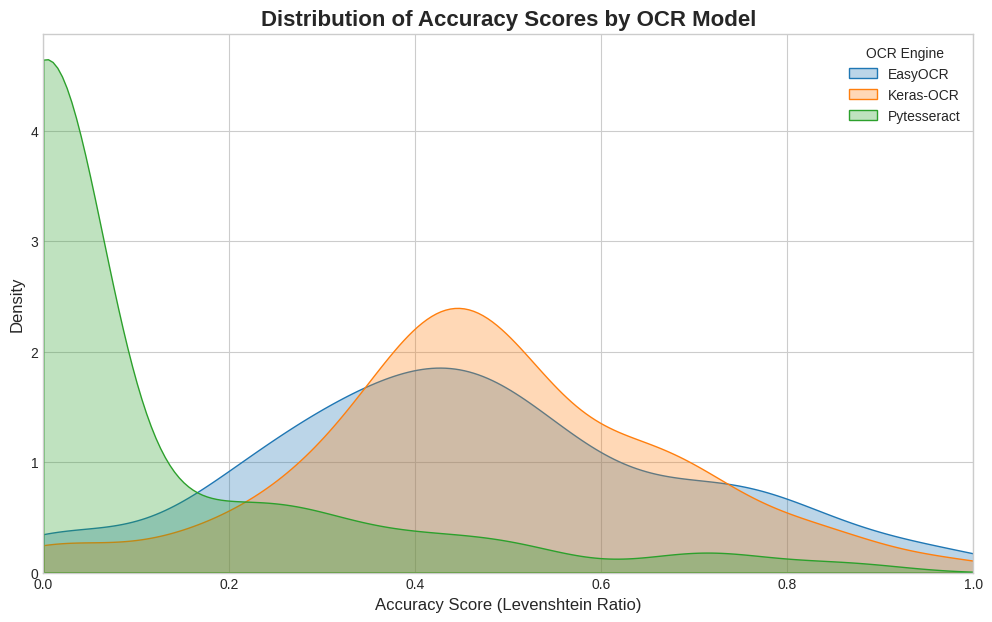
The analysis of the 300-image sample yielded distinct performance levels across the three models. The average accuracy, as calculated by the mean Levenshtein Ratio, is summarized in Table 1.

Table 1 AVERAGE ACCURACY

|  |  |
| --- | --- |
| **OCR Model** | **Average Accuracy(%)** |
| EasyOCR | 45.47% |
| KerasOCR | 47.56% |
| Pytesseract | 10.10% |

*Figure 2. Accuracy Chart*

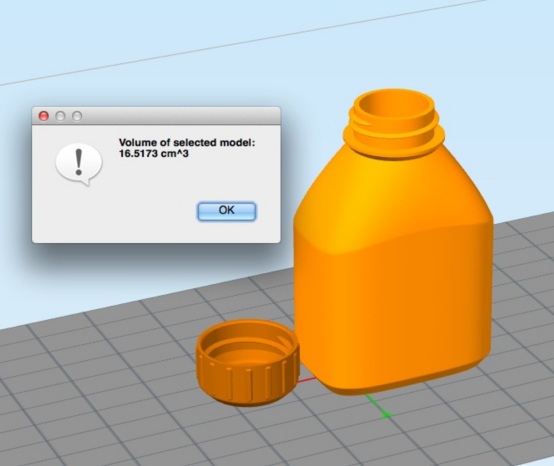
*Figure 3. Number of Times Each OCR Engine Achieved Highest Accuracy*

*Figure 4. Distribution of Each Model*

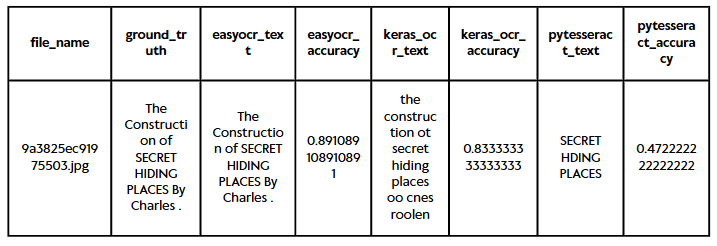
The full, detailed results for each of the 300 images, including the predicted text from each model and its corresponding accuracy score, were compiled and saved in the ocr\_comparison\_results.csv file.

**4.2. Qualitative Examples**

To provide a more nuanced understanding of model performance, specific examples were examined.

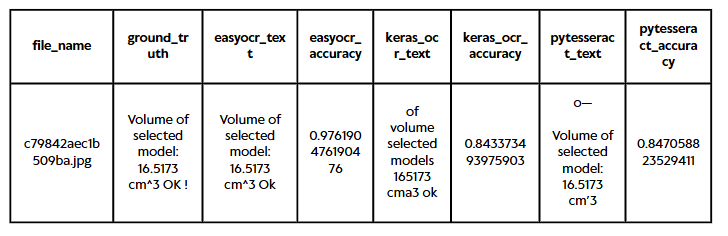
*****Figure 5. c79842aec1b509ba.jpg*

This figure showcases an instance where all models performed with high accuracy. The image contains clear, well-lit text, which presented little challenge to any of the models. The outputs from all three were nearly identical to the ground truth.

Table 2: Accuracy of Each Model

*Figure 6. 9a3825ec91975503.jpg*

In contrast, this figure illustrates a more challenging scenario, such as text on a curved or cluttered background. In this case, Pytesseract failed to read the rest of the text while easyOCR produced a highly accurate transcription closer to the ground truth. Then comes keras-OCR in the second place. This highlights a key difference in the models' capabilities when faced with non-ideal conditions.

Table 3 Accuracy of Each Model

1. **DISCUSSION**

This section interprets the results presented above, contextualizes them within the broader field, acknowledges the study's limitations, and suggests avenues for future research.

**5.1 Interpretation of Results**

The results indicate a clear performance hierarchy among the evaluated models for this specific task. The data from Table I shows that Keras-OCR emerged as the most effective tool, achieving an average accuracy of 47.56%. The superior performance of Keras-OCR can likely be attributed to the robustness of its CRAFT text detector, which was more successful at identifying text regions in cluttered backgrounds and real life image compared to the other models. Same can be said to EasyOCR where it shown a slight difference on the average accuracy. Pytesseract, on the other hand, frequently struggled with identifying text based on real life pictures which tends to lead to nothing being identified leading to its lower overall score. Pytesseract struggles with picture in real life but it works well in a normal environment type picture such as figure 6.

These findings align with established trends in OCR research. Lin et al. (2020) conducted a comprehensive survey of scene text detection and recognition methods, showing the advantages of deep learning based approaches in handling complex, real world scenarios [22]. Moreover, recent advancements such as CLIP-OCR demonstrate that integrating visual-linguistic priors can significantly improve recognition accuracy in scene text recognition tasks [23].

**5.2. Limitation of the Study**

Several limitations should be noted. The study was conducted on a sample size of 300 images, which, while diverse, may not capture all the potential challenges present in the full TextOCR dataset [24]. Furthermore, the dataset primarily consists of incidental English text; these results may not be generalizable to other domains such as scanned historical documents, forms with structured data, or other languages. Finally, this experiment did not explore the potential impact of image pre-processing techniques, which could have improved the performance of certain models. This consideration is supported by findings from [25], demonstrated that performance of the model in analysis can be boosted through semi supervised techniques and input refinement strategies.

**5.3. Future Work**

Building upon this study, future work could validate these findings on a significantly larger portion of the dataset. A valuable next step would be to examine the effects of pre-processing techniques, such as contrast enhancement and denoising, on model performance. Additionally, broader benchmarking against other state of models such as those surveyed by Lin et al. (2020) and more recent deep learning frameworks like CLIP-OCR (Wang et al., 2023)can provide a more comprehensive understanding of scene text recognition performance [22], [23].

**5.4 Data Availbality**

The data used for this study are openly available at: https://github.com/jelllllllllll/Text-Detection-Project

1. **REFERENCES**

[1] M. Li *et al.*, “Trocr: Transformer-based optical character recognition with pre-trained models,” in *Proceedings of the AAAI conference on artificial intelligence*, 2023, pp. 13094–13102.

[2] L. Hamdi, A. Tamasna, P. Boisson, and T. Paquet, “VISTA-OCR: Towards generative and interactive end to end OCR models,” *arXiv Prepr. arXiv2504.03621*, 2025.

[3] L. Biswal, V. Malap, S. K. Chaudhary, and S. Patro, “Impact of Peer Learning on Performance and Motivation in Management Professional in Higher Education Sector-An Empirical Study,” *J. Informatics Educ. Res.*, vol. 4, no. 1, 2024.

[4] W. Khallouli, M. S. Uddin, A. Sousa-Poza, J. Li, and S. Kovacic, “Leveraging Transformer-Based OCR Model with Generative Data Augmentation for Engineering Document Recognition,” *Electronics*, vol. 14, no. 1, p. 5, 2024.

[5] Z. Liu *et al.*, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10012–10022.

[6] S. Ozkan *et al.*, “Efficient and Accurate Scene Text Recognition with Cascaded-Transformers,” in *Proceedings of the 16th ACM Multimedia Systems Conference*, 2025, pp. 335–341.

[7] M. Nazeem, R. Anitha, and S. Navaneeth, “Open-Source OCR Libraries: A Comprehensive Study for Low Resource Language,” in *Proceedings of the 21st International Conference on Natural Language Processing (ICON)*, 2024, pp. 416–421.

[8] A. Kae, G. Huang, C. Doersch, and E. Learned-Miller, “Improving state-of-the-art OCR through high-precision document-specific modeling,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, 2010, pp. 1935–1942.

[9] B. Shaw and D. Mishra, “Character Detection: An Approach to Clarify the Texts Using Machine Learning,” *Procedia Comput. Sci.*, vol. 259, pp. 346–355, 2025.

[10] S. Eken, H. Menhour, and K. Köksal, “DoCA: a content-based automatic classification system over digital documents,” *IEEE Access*, vol. 7, pp. 97996–98004, 2019.

[11] A. Beshirov, M. Dobreva, D. Dimitrov, M. Hardalov, I. Koychev, and P. Nakov, “Post-OCR text correction for Bulgarian historical documents,” *Int. J. Digit. Libr.*, vol. 26, no. 1, p. 4, 2025.

[12] V. K. Soni, V. Shukla, S. R. Tandan, A. Pimpalkar, N. K. Nema, and M. Naik, “Performance Evaluation of Efficient and Accurate Text Detection and Recognition in Natural Scenes Images Using EAST and OCR Fusion,” *Perform. Eval.*, vol. 16, no. 1, 2025.

[13] E. Li, L. Zeng, Z. Zhou, and X. Chen, “Edge AI: On-demand accelerating deep neural network inference via edge computing,” *IEEE Trans. Wirel. Commun.*, vol. 19, no. 1, pp. 447–457, 2019.

[14] J. Martínek, L. Lenc, and P. Král, “Building an efficient OCR system for historical documents with little training data,” *Neural Comput. Appl.*, vol. 32, pp. 17209–17227, 2020.

[15] A. Rahali and M. A. Akhloufi, “End-to-end transformer-based models in textual-based NLP,” *Ai*, vol. 4, no. 1, pp. 54–110, 2023.

[16] R. Pal, S. Kar, and A. A. Sekh, “Customizable and Programmable Deep Learning,” in *International Conference on Pattern Recognition*, Springer, 2024, pp. 101–116.

[17] S. Mukherjee, H. Tyagi, P. Tyagi, N. Singh, and S. Bhardwaj, “OCR Using Python and Its Application.,” *J. Adv. Zool.*, vol. 44, 2023.

[18] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, “Character region awareness for text detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9365–9374.

[19] D. Sporici, E. Cușnir, and C.-A. Boiangiu, “Improving the Accuracy of Tesseract 4.0 OCR Engine Using Convolution-Based Preprocessing,” *Symmetry (Basel).*, vol. 12, no. 5, p. 715, May 2020, doi: 10.3390/sym12050715.

[20] D. Van Strien, K. Beelen, M. C. Ardanuy, K. Hosseini, B. McGillivray, and G. Colavizza, “Assessing the impact of OCR quality on downstream NLP tasks,” 2020.

[21] T. Petty, J. Hannig, T. I. Huszar, and H. Iyer, “A New string edit distance and applications,” *Algorithms*, vol. 15, no. 7, p. 242, 2022.

[22] H. Lin, P. Yang, and F. Zhang, “Review of Scene Text Detection and Recognition,” *Arch. Comput. Methods Eng.*, vol. 27, no. 2, pp. 433–454, Apr. 2020, doi: 10.1007/s11831-019-09315-1.

[23] Z. Wang, H. Xie, Y. Wang, J. Xu, B. Zhang, and Y. Zhang, “Symmetrical linguistic feature distillation with clip for scene text recognition,” in *Proceedings of the 31st ACM international conference on multimedia*, 2023, pp. 509–518.

[24] A. Singh, G. Pang, M. Toh, J. Huang, W. Galuba, and T. Hassner, “Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 8802–8812.

[25] I. Ehsan, T. Shehzadi, D. Stricker, and M. Z. Afzal, “End-to-end semi-supervised approach with modulated object queries for table detection in documents,” *Int. J. Doc. Anal. Recognit.*, vol. 27, no. 3, pp. 363–378, 2024.